CLAIM AMENDMENTS

Claim Amendment Summary

Claims pending

- Before this Amendment: Claims 1, 2, 4-9, 11-18, 20-28, 30-41, 44, 45, 47-62, 64-68, 70-80 and 82-90.
- After this Amendment: Claims 1, 2, 4-5, 7-9, 11-18, 20-26, 28, 30-35, 37-41, 44, 45, 47-62, 64-68, 70-79, 82-87 and 89-90.

Canceled claims: Claims 6, 27, 36, 80 and 88.

Amended claims: Claims 1, 7, 15, 21, 33, 41, 47, 52, 58, 66, 77 and 84.

New claims: None.

Claims:

1. (Currently Amended) A method comprising:

receiving a request, from an application at an application programming interface (API), to interact with a plurality of media; and

generating a media timeline based on the request, wherein the media timeline:

is exposed to the application via the API;

includes a plurality of nodes; and

defines a presentation, to be output via one or more computers, of a first media referenced by a first node with respect to a second media referenced by a second node, wherein:

the first and second nodes are configured as parallel nodes such that the first node that is a child of a parent node is rendered concurrently with the second node that is a child of the same parent node; [[and]]

the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is being rendered; and

at least one node includes metadata, the metadata describing:

rendering of the at least one node; and

a collection of additional nodes to be dynamically modified when the media timeline is rendered.

2. (Previously Presented) A method as described in claim 1, wherein one or more nodes are configured as a sequence node such that one node that is a child of the sequence node is rendered after another node that is also a child of the sequence node.

3. (Canceled)

4. (Previously Presented) A method as described in claim 1, wherein one or more nodes is configured as a root node that specifies a starting point for rendering the media timeline.

5. (Previously Presented) A method as described in claim 1, wherein the first and second said nodes reference the respective first and second media utilizing respective first and second pointers.

6. (Canceled)

7. (Currently Amended) A method as described in ~~claim 6~~ claim 1, wherein the metadata is selected from a group of metadata, the group of metadata comprising :

a URL property for the media referenced by the at least one node;

a source object property that specifies a source object which can resolve to a media source that provides the media referenced by the at least one node;

a source object ID property that specifies a unique identifier of the source object;

a start time property that specifies when rendering of the at least one node is to begin with respect to another node;

a stop time property that specifies when rendering of the at least one node is to stop with respect to another node;

a media start property that specifies a time, during a duration of the media referenced by the at least one node, that rendering of the media is to be started;

a media stop property that specifies a time, during a duration of the media referenced by the at least one node, that rendering of the media is to be stopped;

a time format property that specifies a time format for at least one of the start time property, the stop time property, the media start property, and the media stop property;

a stream selection property which specifies one of a plurality of streams for rendering of the media referenced by the at least one node;

a format based property that specifies a format for the media referenced by the at least one node;

a loop count property that specifies a number of times the at least one node is to be rendered;

a disabled property that specifies whether the at least one node is to be rendered when the media timeline is rendered;

a generic property that serves as a repository of information related to the at least one node, wherein the generic property is configured for specification by at least one of the application and a timeline source for rendering the media timeline;

a noskip property that specifies that the rendering of the at least one node is not to be skipped when the media timeline is rendered; and

a noskip child property that specifies that the at least one node has another node, which is a child of the at least one node, which specifies that the rendering of the other node is not to be skipped when the media timeline is rendered.

8. (Previously Presented) A method as described in claim 1, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

9. (Previously Presented) A method as described in claim 1, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

10. (Canceled)

11. (Previously Presented) A method as described in claim 1, wherein at least one node is specified as read-only.

12. (Previously Presented) A method as described in claim 1, wherein at least one node is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is rendered.

13. (Previously Presented) A method as described in claim 1, wherein the first and second media have different formats.

14. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 1.

15. (Currently Amended) A method comprising:

generating a media timeline by an application, wherein the media timeline:

includes a plurality of nodes;

defines a presentation of a first media referenced by a first node with respect to a second media referenced by a second node, the presentation being configured to be output by one or more computers;

passing the media timeline to a timeline source for rendering; and

is configured for dynamic creation such that at least a first node grouping ~~one node~~ is created while a second node grouping in the media timeline is being rendered.

16. (Previously Presented) A method as described in claim 15, wherein the first and second media have different formats.

17. (Previously Presented) A method as described in claim 15, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

18. (Previously Presented) A method as described in claim 15, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

19. (Canceled)

20. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 15.

21. (Currently Amended) A method <u>for outputting a media presentation via one or more computers</u> comprising:

specifying an effect to be applied to one or more of a plurality of media when the media is rendered; and

generating a media timeline configured for exposure via an application programming interface (API), wherein:

the media timeline includes a plurality of nodes;

two or more of the plurality of nodes reference respective media;

one or more of the plurality of nodes that reference the one or more of the plurality of media include metadata that describes the effect; [[and]]

the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered; and

at least one node includes metadata, the metadata describing:

rendering of the at least one node; and

a collection of additional nodes to be dynamically modified when the at least one node is rendered.

22. (Original) A method as described in claim 21, wherein the effect is a simple effect provided by a software component that is configured to:

receive a single stream of media;

apply the effect to the single stream; and

output the applied single stream.

23. (Original) A method as described in claim 21, wherein the effect is a composite effect provided by a software component that is configured to:

receive at least two streams of media;

apply the effect to the at least two streams; and

output a single stream of media composed of the applied at least two streams.

24. (Previously Presented) A method as described in claim 21, wherein the effect is a composite effect provided by a software component that is configured to analyze at least two streams of media or output at least two streams of media.

25. (Previously Presented) A method as described in claim 21, wherein the effect is a transition effect to be applied as a transition from a first media referenced by a first node to a second media referenced by a second node.

26. (Previously Presented) A method as described in claim 21, wherein the effect includes metadata that describes the effect that is selected from the group of metadata, the group of metadata comprising :

an effect object GUID property that specifies a GUID to be used to create a transform object that is configured to provide the effect;

an effect object property that references an effect object that is configured to provide the effect;

a priority property that specifies an ordering of a plurality of effects, one to another;

a start time property that specifies when processing of the effect is to begin with respect to rendering one of the nodes;

a stop time property that specifies when processing of the effect is to stop with respect to rendering one of the nodes;

a time format property that specifies a time format for at least one of the start time property and the stop time property

a number of inputs property that specifies a number of inputs to the effect;

a number of outputs property that specifies a number of outputs from the effect;

an output major type property that specifies a major type for media, to which, the effect is to be applied; and

an input connections property that specifies the one or more nodes that are to be processed by the effect.

27. (Canceled)

28. (Previously Presented) A method as described in claim 21, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

29. (Canceled)

30. (Previously Presented) A method as described in claim 21, wherein at least one node is specified as read-only.

31. (Previously Presented) A method as described in claim 21, wherein at least one node is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is rendered.

32. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, cause the computer to perform the method of claim 21.

33. (Currently Amended) In a media timeline exposed via an application programming interface and having a plurality of nodes, a method comprising:

rendering a first media item referenced by a first node;

receiving a call for a second node that references a second media item; [[and]]

creating the second node while rendering the first media item; and

wherein the media timeline is configured for dynamic updating such that metadata included in at least one node specifies a collection of nodes to be modified when the at least one node is loaded.

34. (Previously Presented) A method as described in claim 33, further comprising rendering a second media item referenced by the second node when the rendering of the first media item is completed.

35. (Previously Presented) A method as described in claim 33, further comprising:

rendering the second media item referenced by the second node;

receiving a call for a third node that references a third media item; and

creating the third node.

36. (Canceled)

37. (Previously Presented) A method as described in claim 33, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

38. (Previously Presented) A method as described in claim 33, wherein at least one node is specified as read-only.

39. (Previously Presented) A method as described in claim 33, wherein at least one node is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is rendered.

40. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 33.

41. (Currently Amended) In a media timeline exposed via an application programming interface, the media timeline having a plurality of nodes, at least two of which reference respective media, one or more nodes each having metadata that references a node grouping, a method comprising:

utilizing a computer to load ~~loading~~ a first node for rendering;

examining metadata associated with the first node to determine a first node grouping to be loaded in conjunction with the first node;

loading each node referenced by the first node grouping;

rendering the first node grouping;

examining ~~at least one~~ a second node in the first node grouping to determine a second node grouping, wherein the examining ~~at least one~~ the second node in the first node grouping is performed during the rendering of the first node grouping;

loading each node referenced by the second node grouping; and

rendering the second node grouping when the rendering of the first node grouping is completed, wherein:

the media timeline is configured for dynamic creation where [[one]] at least a third node is created while the media timeline is being rendered, the dynamic creation of the [[one]] third node being performed by a node source that includes data that defines properties and interrelationships of the created third node with respect to one or more nodes in the first node grouping or one or more nodes in the second node grouping; and

at least [[one]] a fourth node is configured for communication of an initiated event to ~~another~~ a fifth node which has subscribed to receive events initiated by the fourth node, such that a change ~~may be~~ is made to [[the]] one or more nodes in the media timeline that are affected by the initiated event ~~while the media timeline is being rendered~~, wherein the ~~plurality of~~ one or more nodes of the media timeline that are affected by the initiated event ~~change~~ are ~~automatically~~ dynamically updated.

42. (Canceled)

43. (Canceled)

44. (Previously Presented) A method as described in claim 41, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

45. (Previously Presented) A method as described in claim 41, wherein at least one node is specified as read-only.

46. (Canceled)

47. (Currently Amended) A method as described in claim 41, wherein a ~~first node references media having a~~ plurality of different formats are ~~format than media~~ referenced by one or more nodes~~a second node~~.

48. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 41.

49. (Previously Presented) In a media timeline exposed via an application programming interface (API), the media timeline having a plurality of nodes, two or more nodes each referencing respective media, a method comprising:

rendering a first node to output a referenced first said media;

during the rendering, changing one or more properties of a second node; and

initiating, by an event generator located on the second node, an event for communication to a parent node of the second node, wherein the event describes the changing.

50. (Original) A method as described in claim 49, wherein the event is communicated to at least one of an application over the API and a timeline source for rendering the media timeline.

51. (Original) A method as described in claim 49, wherein the one or more properties are selected from the group consisting of:

node added event;

node removed event;

node changing event;

node changed event;

remove children event;

node source added event;

node source removed event;

node sort event; and

node moved event.

52. (Currently Amended) A method as described in claim 49, wherein:

[[one]] at least one node of the media timeline is configured as a root node; and

each event generated by one of the plurality of nodes that is a child of the root node is communicated to the root node.

53. (Previously Presented) A method as described in claim 49, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

54. (Previously Presented) A method as described in claim 49, wherein the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered.

55. (Previously Presented) A method as described in claim 49, wherein at least one node is specified as read-only.

56. (Previously Presented) A method as described in claim 49, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

57. (Previously Presented) One or more computer readable media storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 49.

58. (Currently Amended) An application programming interface embodied on a computer storage medium, which when interfaced with a computer, exposes a media timeline to one or more independent applications, the application programming interface comprising:

the media timeline comprising a plurality of nodes callable by one application, wherein:

each node includes metadata that describes the node;

one or more nodes reference a corresponding media item;

the plurality of nodes are arranged in a tree structure; and

the arrangement of the plurality of nodes, one to another, describes an order for rendering the plurality of nodes, wherein the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered and at least one node is dynamically updated in response to the at least one node being created.

59. (Previously Presented) An application programming interface as described in claim 58, wherein the metadata for each node is selected from a group of metadata, the group of metadata comprising:

a URL property for the media referenced by the node;

a source object property that specifies a source object which can resolve to a media source that provides the media referenced by the node;

a source object ID property that specifies a unique identifier of the source object;

a start time property that specifies when rendering of the node is to begin with respect to another node;

a stop time property that specifies when rendering of the node is to stop with respect to another node;

a media start property that specifies a time, during a duration of the media referenced by at the node, that rendering of the media is to be started;

a media stop property that specifies a time, during a duration of the media referenced by the node, that rendering of the media is to be stopped;

a time format property that specifies a time format for at least one of the start time property, the stop time property, the media start property, and the media stop property;

a stream selection property which specifies one of a plurality of streams for rendering of the media referenced by the node;

a format based property that specifies a format for the media referenced by the node;

a loop count property that specifies a number of times the node is to be rendered;

a disabled property that specifies whether the node is to be rendered when the media timeline is rendered;

a noskip property that specifies that the rendering of the node is not to be skipped when the media timeline is rendered; and

a noskip child property that specifies that the node has another node, which is a child of the node, which specifies that the rendering of the other node is not to be skipped when the media timeline is rendered.

60. (Previously Presented) An application programming interface as described in claim 58, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

61. (Previously Presented) An application programming interface as described in claim 58, wherein at least one node includes metadata that describes rendering of the at least one node.

62. (Previously Presented) An application programming interface as described in claim 58, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

63. (Canceled)

64. (Previously Presented) An application programming interface as described in claim 58, wherein at least one node is specified as read-only.

65. (Previously Presented) An application programming interface as described in claim 58, wherein at least one node is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is rendered.

66. (Currently Amended) An application programming interface stored on a computer storage medium, that when accessed by a computer facilitates acts comprising:

exposing a media timeline to one or more independent applications, the media timeline comprising a plurality of nodes callable by one application, wherein:

two or more of the nodes reference respective media;

the plurality of nodes are arranged in a hierarchy to include a parent node and a child node; and

the child node is configured for initiating an event for communication to the parent node, wherein the event:

is configured such that a change may be made to one or more properties of the child node while the media timeline is rendered; and

describes the change such that additional nodes associated with the child node are dynamically updated in accordance with the communicated event.

67. (Previously Presented) An application programming interface as described in claim 66, wherein another node, which is not a parent of the child node, subscribes to the child node to receive the event.

68. (Previously Presented) An application programming interface as described in claim 66, wherein another node subscribes to the child node to receive:

the event initiated by the child node; and

one or more events initiated by children of the child node.

69. (Canceled)

70. (Original) An application programming interface as described in claim 66, wherein the event describes a change made to the media timeline, the event selected from the group consisting of:

node added event;

node removed event;

node changing event;

node changed event;

remove children event;

node source added event;

node source removed event;

node sort event; and

node moved event.

71. (Previously Presented) An application programming interface as described in claim 66, wherein:

one node of the media timeline is configured as a root node; and

each event generated by one of the plurality of nodes that is a child of the root node is communicated to the root node.

72. (Previously Presented) An application programming interface as described in claim 66, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

73. (Previously Presented) An application programming interface as described in claim 66, wherein the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered.

74. (Previously Presented) An application programming interface as described in claim 66, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

75. (Previously Presented) An application programming interface as described in claim 66, wherein at least one node is specified as read-only.

76. (Previously Presented) An application programming interface embodied in an infrastructure layer of a computer that, when interacted with by an application facilitates actions comprising:

exposing a media timeline comprising one or more nodes to the application; and

enabling the application to call any of the one or more nodes, wherein each of the one or more nodes:

references corresponding media;

includes metadata describing one or more properties for rendering the corresponding media; and

includes metadata specifying the node as read-only; and

configuring the media timeline for dynamic creation such that at least one of the one or more nodes is created while the media timeline is being rendered.

77. (Currently Amended) A system comprising:

a plurality of media;

a plurality of applications; and

an infrastructure layer that:

provides an application programming interface (API) for interaction by the plurality of applications with the plurality of media when any application is executed; and

exposes a media timeline, callable by the plurality of applications via the API upon an execution thereof, and that defines a presentation of the plurality of media, wherein the media timeline:

includes a plurality of nodes that each reference respective media; [[and]]

is configured for dynamic creation such that at least one node is created while the media timeline is rendered; and

is configured for dynamic loading such that metadata included in the at least one node created specifies a collection of nodes to be loaded when the media timeline is rendered.

78. (Original) A system as described in claim 77, wherein the media timeline is configured to reference an effect for application to an output of one or more of the plurality of media.

79. (Previously Presented) A system as described in claim 77, wherein:

the media timeline defines a presentation of a first media referenced by a first node with respect to a second media referenced by a second node; and

at least one node includes metadata that describes rendering of the at least one node.

80. (Canceled)

81. (Canceled)

82. (Previously Presented) A system as described in claim 77, wherein at least one node is specified as read-only.

83. (Previously Presented) A system as described in claim 77, wherein at least one said node is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is rendered.

84. (Currently Amended) A computer comprising:

a processor; and

memory configured to maintain:

a plurality of media;

a plurality of applications, wherein each application is configured to request at least one of editing, encoding, and rendering of the plurality of media; and

an infrastructure layer configured to:

provide an application programming interface (API) for interaction by the plurality of applications with the plurality of media; and

expose a media timeline, callable by the plurality of applications via the API, ~~that includes~~ including a plurality of nodes that define a presentation of the plurality of media, wherein the media timeline specifies:

delayed creation of one or more nodes when the media timeline is rendered;

metadata that is utilized by the plurality of applications, wherein the metadata describes:

[[how]] initiating rendering of the plurality of nodes ~~is to be initiated~~;

properties and interrelationships of the plurality of nodes; [[and]]

node types of the plurality of nodes; and

dynamic changes to the media timeline such that a group of nodes affected by a modification to an associated node are automatically updated in accordance with the modification as specified in the properties and interrelationships of the plurality of nodes;

at least one node that is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is being rendered; and

at least one node that is a parallel node that provides simultaneous rendering of at least two child nodes.

85. (Previously Presented) A computer as described in claim 84, wherein the delayed creation further comprises creating the one or more nodes when called by one or more applications.

86. (Previously Presented) A computer as described in claim 84, wherein the media timeline is configured for dynamic loading such that metadata included in at least one node specifies a collection of nodes to be loaded when the media timeline is rendered.

87. (Previously Presented) A computer as described in claim 84, wherein at least one node is configured to reference an effect to be applied to an output of media referenced by the node.

88. (Canceled)

89. (Previously Presented) A method as described in claim 33, wherein the first node and the second node are each selected from a group of node types, the group of node types comprising:

a root node that specifies a starting point for rendering the media timeline, the root node including metadata that describes how rendering is to be initiated;

a leaf node that directly maps to media to be rendered and output, the leaf node including metadata that describes how to retrieve the media;

a sequence node that includes metadata that describes a rendering order of a plurality of leaf nodes to the sequence node; and

a parallel node that includes metadata specifying a plurality of leaf nodes that are rendered simultaneously.

90. (Previously Presented) A method as described in claim 41, wherein the first node is selected from a group of node types, the group of node types comprising:

a root node that specifies a starting point for rendering the media timeline, the root node including metadata that describes how rendering is to be initiated;

a sequence node that includes metadata that describes a rendering order of a plurality of leaf nodes to the sequence node; and

a parallel node that includes metadata specifying a plurality of leaf nodes that are rendered simultaneously.